

**REMARKS/ARGUMENTS**

Claims 14-17 and 26-46 are pending. The claims have not been amended.

Claims 14-17, 26-33, and 35-46 are rejected under 35 U.S.C. § 102(b) as being anticipated by Luc Neumann.

Claim 34 is rejected under 35 U.S.C. § 103(a) as being unpatentable over Luc Neumann in view of Using Oracle Jdeveloper and Business Components for Java with Oracle InterMedia.

It is submitted earnest that the claims as previously presented distinguish over the cited references for the reasons set forth below.

**The Neumann Reference**

The cited portions of Neumann are summarized below as understood, in order to respond to the Section 102 rejection. If the examiner's understanding of Neumann differs, the examiner is respectfully requested to provide a detailed explanation of his interpretation of Neumann.

The Neumann reference relates to mobile devices and describes an architecture to provide interactive applications in mobile devices, including rendering graphics and animation. *Page 3, 1<sup>st</sup> paragraph.* The introductory material in Section 3 describes an architecture for distributing rendering tasks among the mobile device and stationary servers.

Section 3.1 discloses a high-level description of the functional components of their architecture.

Section 3.2 describes partitioning and distribution of rendering tasks. Neumann discloses a quality-based distribution of the rendering tasks, and a scene-based distribution, where rendering tasks are distributed to renderers based on quality or based on scene.

Section 3.3 describes an implementation in the WWW environment, using VRML. The Virtual Reality Markup Language (VRML) is similar to HTML in that it uses a set of tags to describe a 3D space in the same way that HTML uses a set of tags to describe a 2D space. Foreground rendering tasks are performed by the mobile device, while background rendering tasks are distributed to renderers on the network.

Fig. 4 shows how a rendering process is separated into foreground scenes which are processed by the mobile device, background scenes which are distributed to renderers on the network. The figures shows the mobile devices sends a rendering request to a request and task manager. The request and task manager separates the foreground and background scenes and sends them either to the mobile device (foreground rendering tasks) or to renderers (background rendering tasks) to perform the rendering. *Page 7, bottom*. Fig. 5 is described as the integration of their architecture in the structure of the world wide web.

Section 4 and Fig. 6 disclose an object extraction method from VRML scenes. Neumann describes the topic of this section as an “experiment for the adaptation platform presented in the previous section.” *Page 8*. The process is shown in Fig. 6 and explained in Section 4:

1. a connection between the client and the application server is established
2. the URL of a VRML world is sent to the application server
3. the application server obtains the VRML file and parses it to produce a scene graph document
4. the scene graph document is sent to the client
5. a user at the client selects desired elements of the world
6. the client creates a document that describes the selected elements and sends the new document to the application server
7. the application server creates a new VRML world from the document
8. the new VRML world is sent to the client, which then visualizes the results using a VRML browser

#### Section 102 Rejection of Independent Claim 14

Claim 14 substantively recites in part “maintaining at a rendering service a resource pool comprising rendering resources from a previous rendering request” and “comparing rendering resources at the rendering site with rendering resources at the user site.”

Claim 14 further substantively recites “storing generated rendering resources corresponding to previous rendering requests” at the rendering server, and “determining whether to generate a given raw resource into a generated rendering resource based on the comparing.”

- Figs. 3 and 4

Neumann shows in Fig. 4 a request and task manager that receives a rendering request; see also Fig. 3. However, a review of the descriptions for Figs. 3 and 4 does not reveal

that the request and task manager maintains rendering resources from a previous rendering request. As understood, the function of the request and task manager is to separate the rendering request into foreground scenes to be rendered and background scenes to be rendered. Fig. 3 clearly shows subtasks being sent to the mobile client and to renderers. Neumann explicitly explains that Fig. 4 shows “a separation of a rendering process into foreground and background scenes to illustrate the dataflow between the different components. The task for the foreground rendering is done on the mobile client and the background rendering is performed on one or several renderers.” *Page 7 and 8.*

There is no mention of maintaining anything from a previous rendering request. Therefore, neither Fig. 3 nor Fig. 4 shows the recited “maintaining at a rendering service a resource pool comprising rendering resources from a previous rendering request.” For at least this reason the Section 102 rejection of claims 14-17 is believed to be overcome.

In addition, neither Fig. 3 nor Fig. 4 show “comparing rendering resources at the rendering site with rendering resources at the user site.” Neumann simply decomposes a rendering task into subtasks and distributes them, there is no comparing of resources at the rendering site with rendering resources at the user site being done. Since there is no teaching of a comparing step, Figs. 3 or 4 also do not show “determining whether to generate a given raw resource into a generated rendering resource based on the comparing.” For at least this reason the Section 102 rejection of claims 14-17 is believed to be overcome.

Figs. 3 and 4 are about decomposing a rendering task and distributing the resulting subtasks. Neither Fig. 3 nor Fig. 4, therefore, shows “storing generated rendering resources corresponding to previous rendering requests” in the resource pool of the rendering site. For at least this reason the Section 102 rejection of claims 14-17 is believed to be overcome.

- Fig. 5

Fig. 5 is described as the integration of Neumann’s architecture in the structure of the world wide web. Neumann gives no further details beyond what is shown in this figure. Therefore, none of the foregoing recited features of claim 14 are shown in Fig. 5 for the same

reasons as set forth above in the discussion of Figs. 3 and 4, and thus the Section 102 rejection of claims 14-17 is believed to be overcome.

- Fig. 6

Fig. 6 is summarized above, and as best understood, includes an application server receiving a URL to a VRML world. First, it is earnestly submitted that receiving a URL to a VRML world does not constitute the recited “receiving at a rendering service a rendering request” of claim 14. Neumann’s application server simply receives a URL (address information) of a location of a VRML file; there is no rendering performed by the application server, and so there is no “receiving at a rendering service a rendering request.” For at least this reason the Section 102 rejection of claims 14-17 is believed to be overcome.

Continuing then, given that the application server receives a VRML file, it then creates a scene graph which is then sent to the client. The application server then receives selected elements from the scene graph, and then generates a new VRML file which is sent to the client; it is the client that generates a visualization from the new VRML file. The description of Fig. 6 does not teach that the Neumann application server is a rendering service “receiving ... a rendering request” or that the Neumann application server is a rendering service “maintaining ... a resource pool comprising rendering resources from a previous rendering request.” For at least this reason the Section 102 rejection of claims 14-17 is believed to be overcome.

Fig. 6 does not compare resources at the application server with resources at the client. Fig. 6 is about the application server identifying elements in a VRML world, allowing a user at the client to select some of the elements, and the application server then providing a new VRML world based on the selected elements. Fig. 6 does not show “comparing rendering resources at the rendering site with rendering resources at the user site.” Since, Fig. 6 does not show a comparing step, then Fig. 6 further does not show “determining whether to generate a given raw resource into a generated rendering resource based on the comparing.” For at least this reason the Section 102 rejection of claims 14-17 is believed to be overcome.

Section 102 Rejection of Independent Claims 26, 37, and 42

Claim 26 substantively recites in part “receiving a rendering request at a rendering service” and “performing at the rendering service a rendering task to produce an image.” Claim 26 further substantively recites in part that “if a required rendering resource is not already stored in a data store local to the rendering server computer system, then uploading that required rendering resource from the user site” and “if a required rendering resource is already stored in the local data store, then obtaining that required rendering resource from the local data store.” See also independent claims 37 and 42, which recite similar features.

- Figs. 3 and 4

Neumann shows in Fig. 4 a request and task manager that receives a rendering request; see also Fig. 3. However, a review of the descriptions for Figs. 3 and 4 clearly reveals that the request and task manager separates the rendering request into foreground scenes to be rendered and background scenes to be rendered sends the resulting subtasks to the mobile client and to renderers for rendering. Neumann explicitly explains that Fig. 4 shows a render and task manager that receives a rendering request and performs “a separation of a rendering process into foreground and background scenes to illustrate the dataflow between the different components. The task for the foreground rendering is done on the mobile client and the background rendering is performed on one or several renderers.” *Page 7 and 8*. Thus, to the extent that the examiner relies on Figs. 3 and 4 to assert that Neumann’s resource and task manager corresponds to the recited “rendering service,” Neumann does NOT teach “receiving a rendering request at a rendering service” and “performing at the rendering service a rendering task to produce an image” because Neumann’s resource and task manager does NOT perform rendering. For at least this reason the Section 102 rejection of independent claims 26, 37, and 42 and their respective dependent claims are overcome.

Moreover, since Neumann’s application server does not perform rendering, then the application server does not show that “if a required rendering resource is already stored in the local data store, then obtaining that required rendering resource from the local data store.”

Neumann’s strategy for visualizing a subset of a VRML file as shown in Fig. 6, involves the application server receiving a VRML file, creating a scene graph which is then sent


to the client, the application server receiving selected elements from the scene graph, and then generating a new VRML file which is sent to the client. There is no discussion of the application server determining that a resource is missing and then obtaining it from the client; the application server simply receives a list of user-selected elements from the client. Neumann does not show that "if a required rendering resource is not already stored in a data store local to the rendering server computer system, then uploading that required rendering resource from the user site." For at least this reason the Section 102 rejection of independent claims 26, 37, and 42 and their respective dependent claims are overcome.

**CONCLUSION**

In view of the foregoing, Applicants believe all claims now pending in this Application are in condition for allowance. The issuance of a formal Notice of Allowance at an early date is respectfully requested.

If the Examiner believes a telephone conference would expedite prosecution of this application, please telephone the undersigned at 650-326-2400.

Respectfully submitted,

  
George B. F. Yee  
Reg. No. 37,478

TOWNSEND and TOWNSEND and CREW LLP  
Two Embarcadero Center, Eighth Floor  
San Francisco, California 94111-3834  
Tel: 650-326-2400  
Fax: 415-576-0300  
GBFY:cmm  
60694498 v1